R Notebook

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*. `

Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

## Load packages  
library(dplyr)  
library(Seurat)  
library(patchwork)  
library(ggplot2)

Read data from GSE123813 (pape: clonal replacement of tumor-specific T cells following PD-1 blockade) <https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE123813>

## Read data  
BCCIO.read <- read.table("C:\\Users\\Ryan\\Downloads\\GSE 123813\\bcc\_scRNA\_counts.txt", row.names = 1, sep="\t", as.is=T)  
  
## Read metadata  
BCCIO.meta.data.readall <- read.table("C:\\Users\\Ryan\\Downloads\\GSE 123813\\bcc\_all\_metadata.txt", row.names = 1, sep="\t", as.is=T)  
  
## Format metadata  
BCCIO.meta.data.read\_colnamesall <- BCCIO.meta.data.readall[1,] # Store column names   
  
BCCIO.meta.data.read\_colnames <- BCCIO.meta.data.read\_colnamesall[,1:3] # Keep only patient, treatment and sort columns  
  
BCCIO.meta.data <- BCCIO.meta.data.readall[2:nrow(BCCIO.meta.data.readall),1:3] # Get rid of column names (first row) and the columns I don't want  
  
names(BCCIO.meta.data) <- BCCIO.meta.data.read\_colnames # Change the column names

Create Seurat object

memory.limit(size = 16000000) #Increase memory limit for vector sizes   
BCCIO <- CreateSeuratObject(counts = BCCIO.read, project = "BCCIO", min.cells = 3, min.features = 200, meta.data = BCCIO.meta.data)

Pre-processing workflow

BCCIO[["percent.mt"]] <- PercentageFeatureSet(BCCIO, pattern = "^MT-") # Store mitochondrial QC metrics in an additional column  
BCCIO <- NormalizeData(BCCIO, normalization.method= "LogNormalize", scale.factor = 10000) # Normalize feature expression   
  
BCCIO <- FindVariableFeatures(BCCIO, selection.method = "vst", nfeatures = 2000) # Feature selection (highly variable features)  
  
all.genes <- rownames(BCCIO)  
BCCIO <- ScaleData(BCCIO, features = all.genes) # Scale data in preparation for PCA   
BCCIO <- RunPCA(BCCIO, features = VariableFeatures(object = BCCIO)) # Perform PCA

Checkpoint (Save Work Here)

save(BCCIO, file="C:\\Users\\Ryan\\Downloads\\GSE 123813\\BCCIO")  
save(BCCIO.meta.data,BCCIO.read, file = "C:\\Users\\Ryan\\Downloads\\GSE 123813\\BCCIOrawdata")

Test out Clustering (Varying Resolution Levels) – Higher resolution = greater clustering –>

BCCIO <- FindNeighbors(BCCIO, dims = 1:10)  
  
## Start with r = 0.3 (overclustered)  
  
BCCIO <- FindClusters(BCCIO, resolution = 0.3, algorithm = 3)  
BCCIO.markers <- FindAllMarkers(BCCIO, test.use = 'wilcox', min.pct = 0.25, logfc.threshold = 0.50, max.cells.per.ident = 500)  
top5 <- BCCIO.markers %>% group\_by(cluster) %>% top\_n(n=5, wt=avg\_logFC)  
DoHeatmap(BCCIO,features = top5$gene, size = 3)  
  
## Try r = 0.2 (overclustered)  
  
BCCIO <- FindClusters(BCCIO, resolution = 0.2, algorithm = 3)  
BCCIO.markers <- FindAllMarkers(BCCIO, test.use = 'wilcox', min.pct = 0.25, logfc.threshold = 0.50, max.cells.per.ident = 500)  
top5 <- BCCIO.markers %>% group\_by(cluster) %>% top\_n(n=5, wt=avg\_logFC)  
DoHeatmap(BCCIO,features = top5$gene, size = 3)  
  
## Try r = 0.1 (overclustered)  
  
BCCIO <- FindClusters(BCCIO, resolution = 0.1, algorithm = 3)  
BCCIO.markers <- FindAllMarkers(BCCIO, test.use = 'wilcox', min.pct = 0.25, logfc.threshold = 0.50, max.cells.per.ident = 500)  
top5 <- BCCIO.markers %>% group\_by(cluster) %>% top\_n(n=5, wt=avg\_logFC)  
DoHeatmap(BCCIO,features = top5$gene, size = 3)  
  
## Try r = 0.05 (looks okay)  
  
BCCIO <- FindClusters(BCCIO, resolution = 0.05, algorithm = 3)  
BCCIO.markers <- FindAllMarkers(BCCIO, test.use = 'wilcox', min.pct = 0.25, logfc.threshold = 0.50, max.cells.per.ident = 500)  
top5 <- BCCIO.markers %>% group\_by(cluster) %>% top\_n(n=5, wt=avg\_logFC)  
DoHeatmap(BCCIO,features = top5$gene, size = 3)  
  
## Try r = 0.03 (around the same as 0.05)  
  
BCCIO <- FindClusters(BCCIO, resolution = 0.04, algorithm = 3)  
BCCIO.markers <- FindAllMarkers(BCCIO, test.use = 'wilcox', min.pct = 0.25, logfc.threshold = 0.50, max.cells.per.ident = 500)  
top5 <- BCCIO.markers %>% group\_by(cluster) %>% top\_n(n=5, wt=avg\_logFC)  
DoHeatmap(BCCIO,features = top5$gene, size = 3)

Full DE Testing (UMAP)

BCCIO <- FindClusters(BCCIO, resolution = 0.05, algorithm = 3) ## Set r = 0.05  
BCCIO.markers <- FindAllMarkers(BCCIO, test.use = 'wilcox', min.pct = 0.25, logfc.threshold = 0.50, max.cells.per.ident = Inf) ## Run through all cells  
top5 <- BCCIO.markers %>% group\_by(cluster) %>% top\_n(n=5, wt=avg\_logFC)  
DoHeatmap(BCCIO,features = top5$gene, size = 3) ## Heatmap with all cells  
  
DimPlot(BCCIO, label = T)  
DimPlot(BCCIO, reduction= "umap") ## Create UMAP  
  
labels <- c("")  
BCCIO <- RenameClusters(BCCIO, labels)   
  
DimPlot(BCCIO, reduction="umap") ## Create labeled UMAP